8. Lukas

Program Name: Lukas.java Input File: lukas.dat

Lukas just learned the rules for declaring a variable in Java. His textbook states that the following rules must be followed:

1. Variable names must start with a letter, the underscore (_) character, or $.,
2. The remaining characters must be letters, numbers, or underscores. (Technically, the $ symbol is allowed as well, but it is not recommended to use it—it is intended for names that are automatically generated by tools.) For the purposes of this program, $ will be accepted.
3. You cannot use other symbols such as ? or %, or any symbol located on the digit keys of a standard keyboard. Spaces are not permitted inside names either. You can use uppercase letters to denote word boundaries, as in cansPerPack. This naming convention is called *camel case* because the uppercase letters in the middle of the name look like the humps of a camel.)
4. Variable names are **case sensitive**, that is, canVolume and canvolume are different names.
5. You cannot use **reserved words** such as double or class as names; these words are reserved exclusively for their special Java meanings. These words are:
   abstract, assert, boolean, break, byte, case, catch, char, class, const, continue, default, double, do, else, enum, extends, final, finally, float, for, goto, if, implements, import, instanceof, int, interface, long, native, new, package, private, protected, public, return, short, static, strictfp, super, switch, synchronized, this, throw, throws, transient, try, void, volatile, while
6. The following are literals that you cannot use as variables name in you program:
   true, false, null

Lukas would like to write a program that checks whether a given variable name is in fact valid in Java, can you help him with this?

Input: Input begins with a **single long line** containing all reserved words from Rule 4. Words will be separated by a comma only (no spaces). The **second line of input is an integer N** (1 <= N <= 20), the number of test cases. The following N lines, each contain a variable name that your program will check to determine if the name is valid or not.

Output: For each test case, output Valid or Invalid. Valid if the name is valid name according to the criteria above, Invalid if the name is invalid according to the criteria above.

Sample input:

```
abstract,assert,boolean,break,byte,case,catch,char,class,const,continue,default,double,do,else,enum,extends,fin
al,finally,float,for,goto,if,implements,import,instanceof,int,interface,long,native,new,package,private,protect
ed,public,return,short,static,strictfp,super,switch,synchronized,this,throw,throws,transient,try,void,volatile,
while
10
yes
true
truE
$100
_thisIsAVariable
this_is_a_variable
number#
1switch
switch
dollar$
```

Sample output:

```
Valid
Invalid
Valid
Valid
Valid
Valid
Invalid
Invalid
Invalid
Valid
```